Qualitative Simulation and Intelligent Tutoring Aids for Training in the Operation of Complex Dynamic Systems

T. Govindaraj
Vijay Vasandani

Technical Report CHMSR-91-1

Center for Human-Machine Systems Research
School of Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332-0205
+1 404 894 3873

1991 February 28

Final Report

for

Office of Naval Research
Contract N00014-87-K-0482
1987 June 1 - 1990 December 31

This report was prepared under the Navy Manpower, Personnel, and Training R&D Program of the Office of the Chief of Naval Research under Contract N00014-87-K-0482.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution unlimited
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) CHMSR-91-1	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Center for Human-Machine Systems Research Georgia Institute of Technology 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Cognitive Science Program Office of Naval Research (Code 1142CS)
6c. ADDRESS (City, State, and ZIP Code) School of Industrial and Systems Engineering 765 Ferst Drive Atlanta, GA 30332-0205	7b. ADDRESS (City, State, and ZIP Code) 800 North Quincy Street Arlington, VA 22217-5000
8a. NAME OF FUNDING / SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N00014-87-K-0482
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. 0602233N; PROJECT NO. RM33M20; TASK NO.; WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)

Qualitative Simulation and Intelligent Tutoring Aids for Training in the Operation of Complex Dynamic Systems

12. PERSONAL AUTHOR(S) T. Govindaraj and Vijay Vasandani

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final	FROM 87.6.1 TO 90.12.31	1991 February 28	29

16. SUPPLEMENTARY NOTATION

Supported by the Office of the Chief of Naval Research Manpower, Personnel, and Training R&D Program.

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	
23	02		fault diagnosis; problem solving; training; intelligent tutoring systems; intelligent computer assisted instruction; marine power plants; simulation

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Supervisory control operators of complex dynamic systems must possess thorough knowledge of the systems, and skills to apply this knowledge to maintain system operation. Use of intelligent tutoring systems (ITS) can be economical for training such operators. An ITS can help organize system knowledge and operational information and also provide practice to develop appropriate skills. We have developed and implemented an ITS on an Apple Macintosh II computer for the marine power plant domain. The ITS is comprised of a simulated power plant, the tutor, and mouse-based direct manipulation graphical interfaces. The ITS was experimentally evaluated using Naval ROTC cadets as subjects under three training conditions: (1) simulator only, (2) a passive tutor that provided help upon request and without intervention, and (3) an active tutor with intervention. Performance in all three conditions was analyzed in identical data collection sessions. Performance measures such as percentage of premature and correct diagnosis and percentage of relevant and irrelevant diagnostic tests were used. Experimental results show that a simulator alone is inadequate, whereas a simulator in conjunction with an ITS can help develop efficient troubleshooting skills. However, not all students are equally receptive to every tutoring strategy and provisions must be made in training programs for individual preferences and differences in abilities and styles.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION	
☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ OTIC USERS	Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL Susan E. Chipman	22b. TELEPHONE (Include Area Code) +1 703 696 4318	22c. OFFICE SYMBOL ONR 1142CS

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

Unclassified

ABSTRACT

Supervisory control operators of complex dynamic systems must possess thorough knowledge of the systems, and skills to apply this knowledge to maintain system operation. Use of intelligent tutoring systems (ITS) can be economical for training such operators. An ITS can help organize system knowledge and operational information and also provide practice to develop appropriate skills. We have developed and implemented an ITS on an Apple Macintosh II computer for the marine power plant domain. The ITS is comprised of a simulated power plant, the tutor, and mouse-based direct manipulation graphical interfaces. The ITS was experimentally evaluated using Naval ROTC cadets as subjects under three training conditions: (1) simulator only, (2) a passive tutor that provided help upon request and without intervention, and (3) an active tutor with intervention. Performance in all three conditions was analyzed in identical data collection sessions. Performance measures such as percentage of premature and correct diagnosis and percentage of relevant and irrelevant diagnostic tests were used. Experimental results show that a simulator alone is inadequate, whereas a simulator in conjunction with an ITS can help develop efficient troubleshooting skills. However, not all students are equally receptive to every tutoring strategy and provisions must be made in training programs for individual preferences and differences in abilities and styles.

INTRODUCTION

Human supervisory control

In the operation of complex dynamic systems such as aircrafts and power plants, human supervisory control operators must process vast quantities of information promptly to maintain desirable levels of system performance. Various subsystems of a complex system generate a large amount of information. This information about the system state must be combined with external inputs from the environment and processed promptly. Even though computers and automatic control systems are generally employed to process information in real time, complete automation based on fully autonomous systems is not possible. Human presence is still required to set high level system goals, monitor system states, and intervene and compensate for problems that the automated control systems are unable to handle.

In such supervisory control environments, the effectiveness of the human operator and his ability to identify and integrate relevant information in a timely manner depends upon his knowledge of system operation, his problem solving skills, and the assistance provided by the system in the form of appropriately chosen and processed information. In this report, we describe re-

search aimed at developing problem solving skills via moderate fidelity simulators and intelligent computer aids.

Operator training for problem solving

Problem solving skills in complex dynamic environments are typically acquired by training on actual systems or on simulators after some basic level of domain-specific knowledge has been acquired. Training on actual systems is usually very expensive. Furthermore, it is undesirable, and often impossible, to simulate failure conditions on actual systems. Hence, simulators can be used to achieve the training objectives at a relatively moderate cost.

In our previous study of training, we investigated the levels of simulator fidelity necessary for teaching problem solving skills (Su, 1985). The domain was oil-fired marine steam power plants. We identified three dimensions of fidelity that were important for training: physical, structural, and dynamic. Physical fidelity relates to the appearance of the simulator in relation to the actual system. Structural fidelity refers to the functional relationship between components. Dynamic fidelity concerns the response and behavior of the simulator states with reference to those of the system. We developed a low fidelity simulator that had moderate structural fidelity, but low physical and dynamic fidelity. This simulator, called FAIL, was used in a training program in which problems based on realistic failures were used (Su & Govindaraj, 1986; Govindaraj & Su, 1988). Trainees using FAIL appeared to follow two distinct stages of troubleshooting: hypothesis formation and hypothesis evaluation. During the diagnostic process, they seemed to use symptom knowledge together with hierarchically organized system knowledge in a combination of backward and forward reasoning strategies. Qualitative descriptions of system states along with compiled or chunked knowledge were used for troubleshooting.

Even when simulators have reasonably high levels of fidelity, major portion of the operating costs in simulator training results from the need for expert instructors. If simulators can be made more intelligent, substantial cost reductions can be achieved while increasing training effectiveness. This can be accomplished by developing simulators at moderate levels of fidelity, and integrating intelligent tutors as part of the simulator design. A moderate fidelity marine power plant simulator that we have developed and integrated with an intelligent tutor and experimental results to evaluate its effectiveness for training are described in this report.

Overview

This report begins with a discussion of a qualitative approximation methodology for the design of complex dynamic system simulators and a description of a marine power plant simulator implemented using qualitative approximation. Architecture for an intelligent tutoring system and some implementation details of the tutor are described next. In the section that follows, impor-

tance of interactive interfaces for tutoring and features of the student-tutor interface are discussed. Details of the experiment to evaluate the tutoring system are provided next, followed by a discussion of the results. The report concludes with a summary of accomplishments and observations.

A MARINE POWER PLANT SIMULATOR VIA QUALITATIVE APPROXIMATION

General

In this section, a qualitative approximation methodology for the design of moderate fidelity simulators is described. In simulators using qualitative approximation, the system states are represented by qualitative measures such as "pressure low" and "flow rate has been steadily decreasing". Exact numerical values are not used. The qualitative state representation aids the human operator by eliminating the need to compare observed state values to nominal values. Also, large systems can be simulated with a moderate amount of computational power due to reduced computational requirements. Basic principles of the design methodology were developed during previous research supported by ONR (Govindaraj, 1987). The current effort enhances these principles by extending them to more general dynamic systems and implementing them in a marine power plant simulator.

The design principles of qualitative approximation are illustrated with reference to a marine power plant with approximately 500 components. The resulting simulator has a moderate level of physical fidelity, and fairly high levels of dynamic, structural, and temporal fidelity. Physical fidelity is concerned primarily with the representation of the control and display interface, and ambient conditions such as noise, vibration, temperature, and humidity. Dynamic fidelity refers to the evolution of system states over time, including the accuracy and the proportion of the total number of states represented. Structural fidelity measures the completeness with which the various components and subsystems are represented. Temporal fidelity is, in a strict sense, a subset of dynamic fidelity, in that it is concerned with the sequential order in which events occur and states evolve. Temporal fidelity is especially important in a qualitative representation of dynamic systems, since even though exact state values are not important, a plausible sequence of state evolution is important.

Hierarchies within a dynamic system

The simulator design methodology is based on a hierarchical description of the system. System components are grouped into a number of subsystems based on their function. For instance, an oil-fired steam power plant on a ship is comprised of the following primary subsystems: fuel oil, feedwater, steam, lube oil, and control air. Some components might belong to more than

one subsystem. For example, the condenser is part of the feedwater subsystem as well as the steam subsystem. Components are classified into a number of generic types, which are then broken down into a small number of primitives. A condenser as well as an economizer, therefore, can be classified as heat-exchangers. This is a rather simple arrangement or design of the hierarchy based on the physical nature of components that form the system.

The primitives form the basic units in the qualitative approximation method. The primitives are the simplest form of components performing a single operation or a function, e.g., providing a path for some fluid in the case of a conduit. Primitives defined in this methodology include: conduit, source, sink, heat exchanger, and resistor. A component such as a condenser can be broken down into two sets of sources and sinks, gains and conduits, and a transfer agent. These hierarchical descriptions follow the natural arrangements of various components and subsystems in the real system.

Description and evolution of the system states

The most significant part of the modeling process in simulator design is the qualitative description of the state space. The states are represented as deviations from their nominal values. This technique, commonly used in modeling dynamic systems, is called the perturbation approach. The key difference from traditional applications, however, is that in the simulator design methodology described here, the perturbed states evolve using approximate functional representations rather than exact representations of the primitives.

Each of the primitives has a structure and a set of parameter values. Structures of the primitives are based on approximate functional equations of system dynamics. The structure, characterized by appropriate differential and algebraic equations, is the same for a primitive regardless of the component which it represents. The parameter values depend on the component of which a particular primitive is a part. Parameters associated with the primitives of a component are tuned to maintain temporal fidelity of state evolution.

System state is updated in a two-step process: during the first step, the states of individual components are updated; during the second step, the updated states are propagated to successor components. Numerical values corresponding to deviations from nominal values are used to represent the states in the simulation. Since these numerical state values are derived from functionally approximate system equations, they represent system states only qualitatively. The state values are transformed into qualitative descriptions, e.g., pressure low and level high, before presenting them to the operator.

An approximate, qualitative representation of system states enables the simulator to maintain cognitive compatibility with trainees. The simulator is said to be cognitively compatible with

its human operator when the qualitative states are similar to state descriptions used by the human. Humans often use qualitative descriptions of system states, e.g., pressure is low and temperature is fluctuating, rather than specific values, e.g., the pressure is 1150 psi. The simulator uses similar states; in training, there is no need for an extremely precise numerical state description. Although the simulation evolves qualitatively, temporal fidelity is maintained since the sequence of state changes that occurs as a result of an event is the same as it would be in a real system.

Implementation of marine power plant simulator

Qualitative approximation methodology was used to design a marine power plant simulator (Govindaraj, 1987). The simulator was based on a hierarchical representation of subsystems, components, and primitives together with necessary physical and logical linkages among them. This simulator, called QSTEAM, was implemented on a Xerox 1100 series LISP machine and provided low physical fidelity. High degrees of structural and dynamic fidelity, however, were achieved.

We have implemented the current version of the simulator, called **Turbinia**, in Allegro Common Lisp (now Macintosh Common Lisp) with Common Lisp Object System (CLOS) and runs on Apple Macintosh II family of computers (Vasandani & Govindaraj, 1989, 1990). This new implementation was dictated by our desire to build the entire system on a widely available computer that is also more easily affordable. The basic hierarchical approach, characteristic of qualitative approximation, was retained. CLOS was used to represent all important entities as objects, including knowledge of the domain and troubleshooting knowledge. Modeling the entities as objects results in a very robust, modular implementation that is easy to maintain and evolve. Mouse-based direct manipulation graphical interfaces were used for the simulator and the tutor. Details of the interface are described in a later section. More complete descriptions the entire system are given in Vasandani (1991).

This section described a qualitative approximation methodology used for designing training simulators of complex dynamic systems. A marine power plant simulator was designed using this methodology. An intelligent tutoring system was used along with the simulator for training Naval ROTC cadets to help develop good problem solving skills. Details of the architecture for the intelligent tutoring system are given in the next section.

ARCHITECTURE FOR AN INTELLIGENT TUTORING SYSTEM

Although work on intelligent tutoring systems have been in progress for over two decades, computer power and developments in cognitive science and ITS research have not been suffi-

ciently harnessed for applications in complex dynamic engineering domains. Among the extensive surveys found in Sleeman and Brown (1982), Wenger (1987), and Psotka, Massey, and Mutter, (1988), only a few deal with engineering domains. STEAMER (Hollan, Hutchins, & Weitzman, 1984), IMTS and its successors (Towne & Munro, 1988), AHAB (Fath, Mitchell, & Govindaraj, 1990), and SHERLOCK (Lesgold, Lajoie, Bunzo, & Eggan, 1989) are some examples that have useful applications in an operator training program.

In general, ITSs have an expert module, a student module, and an instructional module (Sleeman & Brown, 1982; Wenger, 1987; and Psotka et al., 1988). In addition a simulator provides the training environment. The expert module of an ITS contains the domain expertise which is also the knowledge to be taught to the student. The student module contains a models of the student's current level of competence. The instructional module is designed to sequence instructions and tasks based on the information provided by the expert and student models. Also, the interface used to communicate the knowledge to the student can be treated as a separate component of the ITS.

We describe below an architecture for an ITS suitable for training operators of complex dynamic systems. Figure 1 illustrates the major components of the instructional system. Together with the simulator and an interactive interface, the three components of the tutor (i.e., the expert, student, and instructional modules) comprise the architecture for the instructional system. The instructional system has two major requirements: (1) a domain simulator and (2) organization of knowledge that supports the functions of the three major elements of the tutoring system. A brief discussion of the simulator preceded this section. Details of knowledge organization are discussed below.

Representation of knowledge for tutoring

Operators of complex dynamic systems, in which interdependent subsystems have some level of autonomy, must be familiar with operational principles of different types of system, e.g., thermodynamics and heat transfer for the fuel system, or electrical characteristics for a turbogenerator. In addition, the operator must know the nominal values of the state variables and parameters. Problem solving and compensation for failures require processing of information from various subsystems using efficient troubleshooting strategies. Therefore, an intelligent tutoring system must be capable of organizing and presenting knowledge about the system and troubleshooting task knowledge at several levels of granularity or detail. Pedagogical knowledge concerning instructional strategies is also necessary. A brief discussion of these three components of knowledge follows.

System knowledge for an intelligent tutor can be represented in several ways. These representations differ in the levels of description or detail depending on the task for which the knowl-

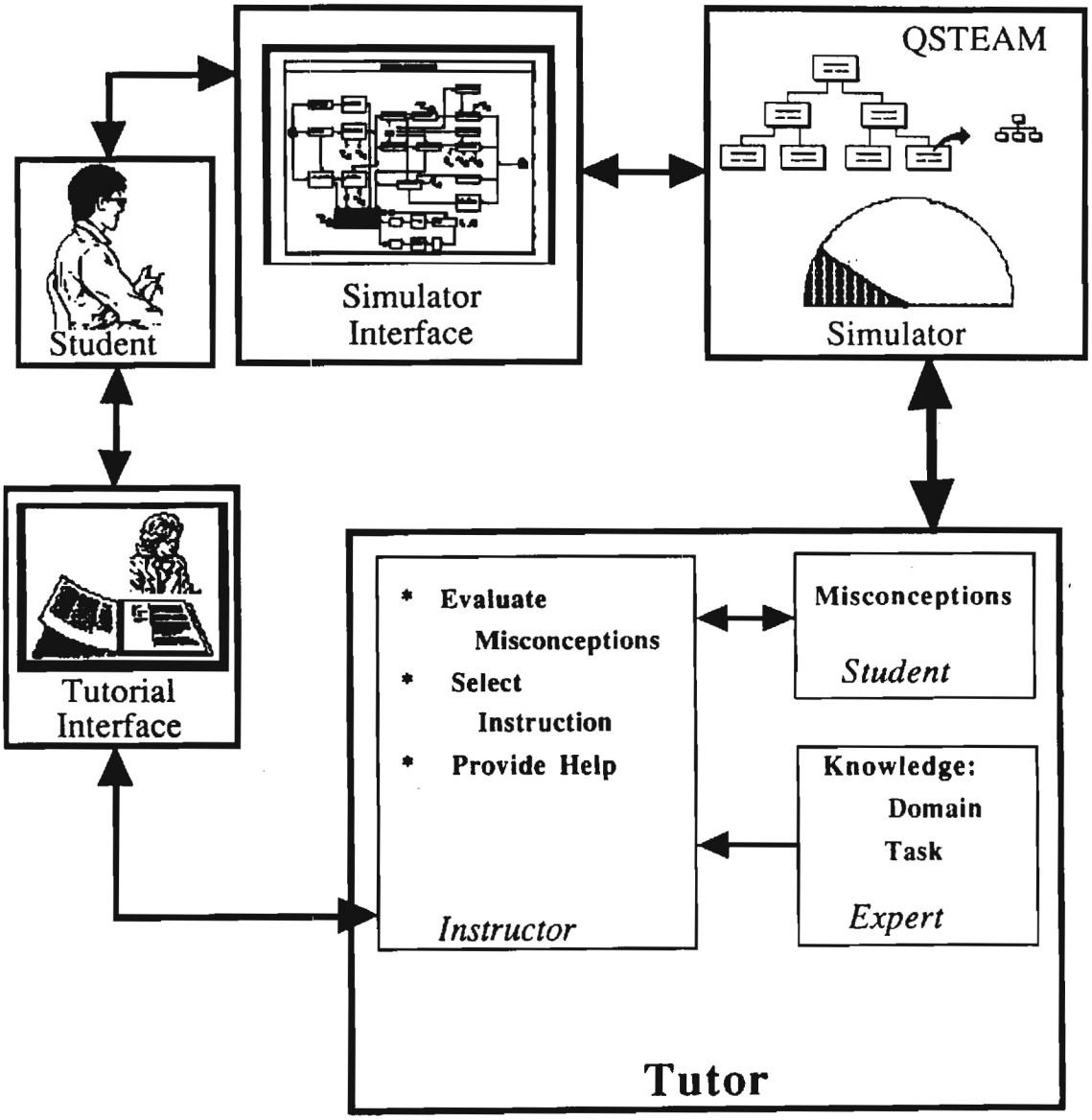

Figure 1. Instructional System

edge is required (Rasmussen, 1986). For most large, complex, mechanical systems, such as steam power plants, a comprehensive representation of system or domain knowledge is via *functional subsystems*, *fluid paths* and *schematics*. A functional subsystem is a collection of components responsible for performing a higher level system function such as combustion in a power plant. Fluid paths are a way of visualizing the system as a collection of different fluids. The fluids may be fuel, steam and air in a steam power plant, or electric current in an electrical circuit. A schematic is a pictorial representation of the system structure through its components and gauges. For large systems, a pictorial representation usually spans several schematics. Although the three representations are complementary rather than mutually exclusive, multiple representations of the system knowledge are necessary to enhance diagnostic performance (Rasmussen, 1986).

Each of the three system representations is an aggregation of mechanical components. System knowledge at the component level concerns a component's *structure*, *function* and *behavior* at a level of detail necessary for successful fault diagnosis. A component's structure refers to its input and output connections to other components, the fluids carried by it, the gauges attached to it, and its association with a functional subsystem. A component's function depends upon its structure and defines the purpose of the component in the system and its contribution to the higher level system functions. The manner in which the system state values are affected by the presence of a component in both the normal and failed states constitutes behavioral knowledge.

System knowledge, although essential, is not adequate for diagnostic problem solving. *Troubleshooting knowledge*, in addition to system knowledge, is equally important. Troubleshooting knowledge includes general information about modes of failure in components, detailed information on certain common failures, cause-effect associations for familiar failures, and behavioral information about components and subsystems under different failure conditions (Govindaraj & Su, 1988; Fath et al., 1990).

Troubleshooting task knowledge, like the system knowledge, can have several representations. However, the representation most suitable for expert performance is not necessarily the best representation for a tutor (Clancey, 1987). The tutor's troubleshooting task knowledge must be organized in a manner that facilitates inference of possible misconceptions based on observed actions. Therefore, expressing troubleshooting task knowledge for each failure in terms of valid student actions at the student-tutor interface is useful for a tutor. Such a representation provides a *normative task model* of troubleshooting and offers a convenient way to interpret and analyze student actions.

Knowledge requirements of an intelligent tutor also include knowledge to evaluate and rectify misconceptions. When a student's actions deviate from the normative task model of troubleshooting, they may be explained in terms of misconceptions related to system structure, func-

tion or behavior. Knowledge that relates a deviant action to a probable misconception and the knowledge to rectify the misconceptions are both essential elements of the instructional system. An efficient way to represent such a knowledge is in the form of rules.

Finally, *pedagogical knowledge* concerning instructional content, form and time of presentation must also be addressed in the design of the instructional system. Instructions can be provided with or without intervention. Instructions with intervention must be provided during the session while those without intervention are provided at the end of a training session. Intervention at critical stages of the training period is an effective way of emphasizing a point; non-intervention has the advantage of not interfering with the thought process of the student.

System, troubleshooting and pedagogical knowledge are important for an instructional system. However, knowledge alone is insufficient to make a tutor effective. It must be appropriately structured and implemented so that the tutor can access relevant pieces of knowledge correctly and promptly as the need arises. Certain details concerning how the knowledge described above is implemented are described next.

Knowledge representation in Turbinia-Vyasa

Turbinia-Vyasa is an instructional system that trains operators to troubleshoot marine power plants. **Turbinia**[1] is the name of the simulated marine power plant used in the instructional system and **Vyasa**[2] is the computer-based tutor that teaches the troubleshooting task using **Turbinia**. This instructional system has been implemented on a dual screen Apple Macintosh II computer in Allegro Common Lisp with object-oriented extensions.

The tutor uses *objects* to encapsulate knowledge about the system and the failures. This knowledge is represented in declarative as opposed to procedural form and is amenable to changes. For example, components in the power plant that are instances of the same functional primitive have similar representations and share *methods* that create and manipulate data. Examples of some abstract data types used for representing knowledge are described below.

> ***System***: System knowledge is decomposed into nine subsystems, thirteen fluid-paths, and seven schematics. Knowledge concerning these functional subsystems, fluid-paths, and schematics is represented in instances of class objects defined for each. Table 1 shows how knowledge concerning combustion subsystem is represented.

---

1. Turbines were first used in marine propulsion by Sir Charles Parsons in 1897 in the *Turbinia*. It was an experimental vessel of 100 tons, fitted with turbines of 2,100 hp driving three propeller shafts. *Turbinia* attained the then record speed of 34.5 knots (Burstall, 1965, p.340).
2. Ancient Indian sage, scholar, and teacher.

Functional-subsystem:

combustion-subsystem is an instance of *subsystem*

subsystem-name	combustion-subsystem
in-schematics	(fuel-oil-schematic boiler-schematic)
fluids	(flue-gas combustion-air fuel-oil steam)
components	(...)
connectors	(...)
function	"To mix the combustion-air with fuel and ignite it in the burner to release thermal energy"
steam-schematic	nil
boiler-schematic	subsystem structure within the schematic components: (...) connectors: (...)
feedwater-schematic	nil
fuel-oil-schematic	subsystem structure within the schematic components: (...) connectors: (...)
control-air-schematic	nil
saltwater-schematic	nil
lube-oil-schematic	nil

Table 1. Description of combustion subsystem

Components: All components are instances of subclasses of either *simple* or *composite primitives*. There are nine subclasses of simple and one subclass of composite primitive defined for the system. There are methods for each subclass of primitives that update the system state across each component.

Failure modes: The system behavior associated with component failure modes is described in instances of *failure-mode* class of objects. There are four failure-mode objects, one for each of the four failure types: blocked-shut, stuck-open, leak-in, and leak-out. Knowledge about a failure mode includes information about the upstream and downstream system behavior along the affected fluid-path.

Failures: Knowledge of the individual faults in the component is stored in objects of class *component-fault*. Knowledge of specific faults includes information concerning affected schematics, subsystems, fluid-paths, and gauges along with explanations of cause-effect associations.

Knowledge concerning evaluation and rectification of student's misconceptions is represented

as rules. A blackboard captures the dynamic evolution of expert and student behavior. More complete details of the tutor implementation are described in Vasandani (1991).

Knowledge organization that captures system structure, function, and behavior and trouble-shooting task knowledge are insufficient for the success of a tutoring system. Properly designed interactive interfaces can also play a major role in imparting knowledge about the system and its operation during normal and abnormal situations. In the next section, we describe the importance of such interfaces and how a student interacts with the instructional system.

INTERACTIVE INTERFACES AND STUDENT-TUTOR INTERACTION

Direct manipulation, interactive, graphical interfaces

A good interface makes the knowledge of the tutor transparent to the student and helps the student understand the complex structure, function, and behavior of the controlled system. In addition, a well designed interface addresses the external-internal task mapping problem (Moran, 1983) and establishes a semantic link between the actions relevant to the task in the domain and the actions to be taken at the interface (Miller, 1988).

In a diagnostic problem solving task, a set of schematics often serve as a convenient student-tutor interface for knowledge communication. These schematics are designed to minimize the external-internal task mapping problem by having the valves and gauges that are usually under the control of the student appear as manipulable objects. Other factors such as grouping of components and graphics also influence the design of these schematics.

Grouping of components in schematics depends upon the degree of logical proximity between components and subsystems; the extent of diagnostic actions necessary to investigate frequent failures; and layouts that ensure smooth transition between schematics. For example, a high degree of interaction between the steam generation and combustion subsystem of a power plant requires that the two subsystems be displayed on the same schematic. Similarly, logically proximate components such as the stack and the burner in a combustion unit must appear together in a schematic. Also, the connections that are discontinued on the left edge of one schematic must continue from the right edge of the connected schematic to maintain visual momentum (Woods, 1984).

Graphics and icons in a schematic interface can enhance the performance of instructional systems (Hollan et al., 1984). Graphical objects or icons can be effectively used to represent meaningful objects or concepts of the system. For instance, in engineering, it is customary to represent a turbine as a trapezoid with the smaller cross section representing the inlet to the tur-

bine. The trapezoidal shape also reminds the viewer that the steam expands in the turbine as it moves from a smaller cross section inlet to a larger cross section outlet. Similarly, concepts such as *blocked-shut* valve and functional subsystems of a power plant can also be represented by meaningful icons.

Schematics provide an interface between the student and the simulated system as well as between the student and the tutor. The tutor uses the schematics to highlight components that constitute a subsystem or share a common fluid path. Such graphical techniques promote visualization of functional subsystems and their interaction. Schematics can also be used by the tutor to animate fault propagation by highlighting gauges as they turn abnormal under simulated failure conditions.

While schematics along with the simulation provide a practice environment that emulates the real system, they do not cover all aspects of student-tutor interaction. For example, the students require a set of expressive techniques to state their hypotheses. The tutor, apart from observing actions, needs a method of seeking information to evaluate misconceptions. Thus, the tutor interface design also involves developing student- and tutor-initiated channels of communication. Since the ability of the instructional system to answer questions is limited by its knowledge and the way this knowledge is organized, the interface must be designed to control and guide the interaction between the student and the tutor.

In student-initiated communications, the interface has to assume the responsibility of guiding the user into asking the right type of questions. For instance, when the student seeks information concerning the system's structure, function, or behavior, it is helpful to make the student select appropriate, context-relevant queries from a set of menus. Such menus, when organized hierarchically, can also reflect the inherent hierarchical structure of the complex system and promote a better understanding of the system (Miller, 1985). Furthermore, an interface that has the provision to address identical queries via multiple representations of the system helps consolidate knowledge from multiple perspectives.

For communications initiated by the tutor, the interface design involves getting the student to understand and correctly respond to the queries. Where a student can respond to a query in multiple ways, the student options have to be recognized in advance and the choice restricted to known alternatives. For example, when the student is asked to provide hypotheses concerning the most likely mode of failure, it makes sense to confine the student's response to only those modes of failure that are known to the tutor. Therefore, making the student select from viable alternatives instead of permitting unguided response is a better approach to interface design.

The issues related to the design of interfaces described above have been implemented in an intelligent tutor for diagnostic problem solving. Implementation details of the student-tutor inter-

face designed to communicate its knowledge to the student are discussed next.

Student-tutor interface and operator interaction

The interface to **Turbinia-Vyasa** consists of seven schematic windows and a large number of dialogs and menus that establish communication between the student and the tutor. A single button computer mouse is the only input device used to interact with this direct manipulation interface.

The seven *schematics* display the physical connections between the components of the power plant. Figure 2 is an example of the boiler schematic. The student can access these schematics through the seven icons displayed in a schematic menu. The schematics are used to investigate components and probe gauges attached to these components. Investigating a component involves moving the cursor on to the component and clicking the mouse button. The same action is also used to indicate diagnosis, pick the component for which information is desired, and indicate hypotheses. Thus, identical student actions on a schematic have different responses. The actual response depends upon the mode of the system in which the interaction occurs. There are four system modes in which interaction occurs: troubleshooting mode, diagnose mode, student-initiated tutor dialog mode and tutor-initiated dialog mode. Except for the tutor-initiated dialog mode, the student is responsible for switching between the remaining three modes of the system. The current mode of the system is indicated by a highlighted icon representing the mode and also by the shape of the cursor.

Student-initiated interaction with the tutor is accomplished by clicking on either the stop icon in the requests menu or a menu item in the hypothesis menu (see Figure 2). This action halts the simulation temporarily, enabling the student to interact with the tutor while preserving the information concerning system states.

When the tutor is invoked using the stop icon in the requests menu, the student can explore the tutor's knowledge-base. The student's exploration is guided by a set of hierarchically organized dialog menus that reflects the inherent system hierarchy. The student can access knowledge concerning specific modes of failure, components, subsystems, and fluid-paths. The information accessed by the student is presented in textual or graphical form.

The interface offers substantial flexibility to the student and maintains context-sensitivity. For example, when the student inquires about fluid-paths following an inquiry about a subsystem, the system first offers a choice of selecting a fluid-path from only among those that can be found in the inquired subsystem. Thus, knowledge about subsystems and fluid-paths does not have to be obtained independently to deduce which fluid-paths lie in which subsystems.

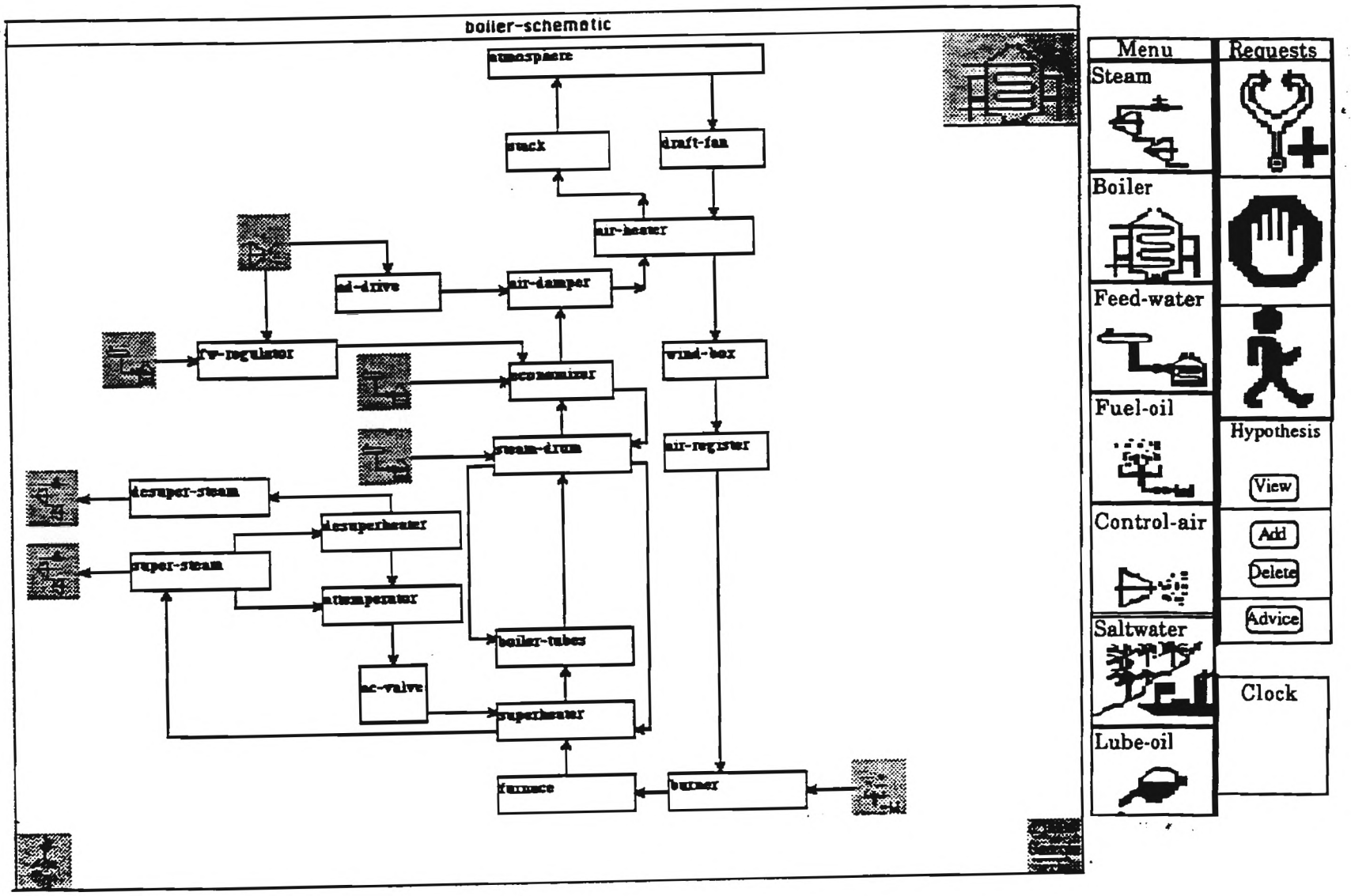

Figure 2. Boiler Schematic and Menus

The student can also invoke the tutor by selecting any of the four items on the *hypothesis menu* (see Figure 2). The first item, "View", is used by the student to review his own hypotheses concerning failure. The "Add" and "Delete" items are used by the student to modify the hypotheses. "Advice" is used to seek assistance from the tutor in refining current hypotheses. The assistance provided is in the form of diagnostic tests that can strengthen or weaken an indicated hypothesis. This helps the student develop an association between failures and its effect on system behavior.

The tutor intervenes to initiate communications with the student in three situations. First, the tutor intervenes when it becomes necessary to notify the student about an error. The error could be an incorrect diagnosis or an invalid action. Second, the tutor intervenes when a possible misconception concerning the system's structure, function or behavior is identified based on the student's actions. When possible misconceptions are identified, the tutor provides canned, but context-sensitive instructions to rectify the student's misconceptions. Third, the tutor sometimes prompts the student for an action or requests for hypotheses concerning failure. In all the three situations, the tutor's communication begins with a beep to draw the student's attention.

In this section we described the importance of interactive interfaces and details concerning student-tutor interaction. Experiments were conducted to evaluate the tutoring system and to determine the benefits of using the instructional system in addition to a simulator for training operators in troubleshooting dynamic systems. Details concerning the experimental evaluation are discussed next.

EXPERIMENTAL EVALUATION OF THE TUTORING SYSTEM

Prior to beginning formal experiments to test the effectiveness of the tutor, the tutor was evaluated with the help of a naval officer who is also an ROTC instructor experienced with steam power plants. This evaluation was informal and somewhat subjective, and was aimed at testing the aiding material and instructional strategies. A set of formal experiments designed to measure the diagnostic performance of operators trained with and without the aid of **Turbinia-Vyasa** followed. The discussion in this section deals with the details of the evaluation.

Checking for consistency and correctness

The primary goal of the subjective evaluation was to ensure that the instructional material presented to the student was technically correct, properly stated and consistent with the current training program for engineers in the US Navy. A secondary goal was to gather suggestions for improving the interface to **Turbinia** and **Vyasa**.

During the subjective evaluation, the experimenter solved problems on **Turbinia-Vyasa** while a subject matter expert, a Naval ROTC instructor, observed the interactive performance feedback from the tutor. The instructor was requested to report any inconsistencies that he observed. He was also asked to make suggestions and comments concerning the design of display and operator interaction. Notes were made of the changes suggested. These notes were later discussed in detail. Following the discussion, several of the suggested changes were incorporated. However, the most useful outcome of this analysis was the reaffirmation of the experimenter's faith in the technical validity of the material presented to the student.

At the conclusion of this evaluation, both the subject matter expert and the experimenter were confident that the students would be receptive to **Turbinia-Vyasa**'s tutoring strategy and that the instructional system would be a worthwhile contribution to the Naval ROTC training program. Formal experiments were started after this evaluation.

Formal experiments

In the formal experiments, performance of subjects trained with and without the tutor was compared. There were two goals in the experiment: (1) determining the effectiveness of the tutoring architecture and methods for knowledge representation and (2) establishing the usefulness of computer-based training programs over traditional means of training operators to troubleshoot complex dynamic systems. In addition, the experiment provided an opportunity for comparing the effect of passive and active tutoring strategies. The above goals can be formulated in terms of the following questions: (1) is it feasible to build an effective computer-based tutor by implementing the proposed architecture, (2) is the training by computer-based tutor better than the training provided by the simulator alone, and (3) how does the level of aiding during the course of training affect performance. Details of the experiment to find answers to these questions are discussed below.

The experiment consisted of two phases: training and data collection. In the training phase, subjects were exposed to one of the three instructional methods: (a) training on simulator alone; (b) training with the aid of a passive tutor; and (c) training with the aid of an active tutor. During data collection, trained subjects from all three conditions attempted to solve the same set of problems unaided by the tutor. A complete description of the experimental design follows.

Experimental design

There were two main factors of interest in the experiment: training condition and seen status of the problem. There were three training conditions: unaided simulator, aiding with passive tutor, and aiding with active tutor. The first was a baseline condition where training was provided using just **Turbinia**. The second and the third training conditions used the computer-based tutor

Vyasa. In the second condition **Vyasa** functioned in the passive mode. It functioned in the active mode in the third condition.

The second factor of interest was the seen status of the problem. We anticipated that this factor would influence performance and therefore should be investigated. Seen status had three levels: seen once, seen twice and unseen. Seen once status applied to problems that were seen once before by the student during training. Similarly, seen twice status applied to problems that were seen twice during training. Unseen status referred to problems that were not seen by the student until the test phase. Since the instructional system was expected to train for not just familiar situations, the effect of seen status was important to analyze the transfer of training from familiar to unfamiliar situations.

Subjects and problems were two other factors that could account for variations in the experimental data. While subjects were nested within training condition, problems were nested within seen status. Subjects and problems along with the main factors and their interactions make a complete list of sources of variation considered in this experiment.

Training condition and seen status were fixed factors and subjects (nested within training condition) and problems (nested within seen status) were random factors. Therefore, a mixed model was used to analyze data from the experiment.

Equipment

For the experiment, **Turbinia** (simulator) and **Vyasa** (tutor) were installed on a dual screen Apple Macintosh II workstation with a 40MHz accelerator board. This computer was used for all data collection sessions. An Apple Macintosh II cx was used for few sessions during initial stages of training. Training instructions for the three experimental conditions were tape-recorded in a female voice on separate tapes to be played back to the subjects during their first training session.

A pilot study

The actual experiment was preceded by a pilot experiment. The purpose of the pilot experiment was to validate the instructional material and to determine software errors that may have gone undetected. The pilot study thus offered the experimenter an opportunity to evaluate the software under experimental conditions.

Four graduate students from Georgia Institute of Technology participated in the pilot study. All four came from engineering backgrounds, had several courses in thermodynamics and were well exposed to research issues related to human-machine systems.

The pilot study was conducted with **Turbinia-Vyasa** operating in the active mode. Since the simulator and the passive tutor were also functional in this mode they were not subjected to separate pilot studies. Each of the four pilot subjects participated in four sessions. In the introductory session, the subjects read the instructional manual to become familiar with the **Turbinia-Vyasa** interface. Then, using the instructions, the subjects attempted to solve a single problem designed specifically for the first session. In the subsequent sessions, each subject saw four problems in every session.

The pilot study identified several discrepancies in the instructional manual and errors in software. These were corrected promptly. Furthermore, five problems out of the twenty-nine supplied by the Navy were eliminated from the set to be used in the full-scale experiment because they were either redundant or exhibited inconsistent behavior. Of the remaining twenty-four problems, those that had software errors were modified, sometimes more than once, and resolved by the pilot subjects till the error was corrected.

Apart from detecting errors in the instructional manual and the software, the pilot study was also useful in estimating time taken to solve problems. This helped the experimenter design the appropriate duration of the introductory as well as subsequent training and data collection sessions.

At the conclusion of the pilot study, the experimenter was better equipped to answer questions from subjects participating in the full-scale experiment. This was important as answers to similar questions by subjects in the three experimental conditions had to be consistent and required advance preparation.

The experimenter could not get an initial estimate of the full-scale experimental results however, since the pilot subjects were not drawn from the same population as that of subjects that were to participate in the full-scale experiment.

Experiment

Thirty cadets from the Georgia Institute of Technology Naval ROTC unit participated as subjects in the experiment. All except one subject were male. Subjects were required to have a basic understanding of the theory of marine power plants. Therefore, sophomores and juniors who had taken the freshman-level course in Naval Engineering offered by Navy ROTC were considered. Among those cadets who volunteered, 24 sophomores and 6 juniors were selected. Although some of the subjects had additional exposure to thermodynamics through course work or had experience operating marine power plants, effects of these factors were not considered. Therefore, the assignment of subjects to the three experimental groups was done randomly.

Each subject was paid $6 per session for every session completed in both the training and the testing phase. In addition, an award of $25 was promised for the best troubleshooter in each of the three groups based on performance in the two data collection sessions.

Subjects were told about the performance measures prior to the experiment. (Performance measures are described in a later section.) They were also informed that the number of problems correctly diagnosed in minimum time was the only measure for the purpose of determining the award.

Experimental materials

There were separate written instructional manuals for each of the three experimental conditions. The manual for subjects using just the simulator (**Turbinia**) provided an introduction to the marine power plant, its automatic boiler control system, a description of the common modes of failure and a guide to make the subjects familiar with **Turbinia**'s interface. For subjects using **Turbinia-Vyasa** in the passive mode additional instructions were included that described the features and the interface of the passive tutor. Further instructions describing the capabilities of the tutor were added to the manual for subjects using **Turbinia-Vyasa** in the active mode.

Other material used in the experiment included a subject consent form and a survey form. The survey form was designed to obtain a feel for the subjects' academic background, shipboard experience and computer skills.

Subjects were required to answer three written questionnaires at three key points in the experiment. The first questionnaire was used to evaluate the subjects' operating knowledge of the marine power plant, its components and their behavior under normal and failed states prior to the start of training. The second questionnaire was identical to the first and was used at the end of the training sessions. Answers to the two questionnaires provided the experimenter with some means to measure the knowledge acquired during training. Subjects answered the third questionnaire at the end of the last data collection session to provide subjective opinions about various aspects of the training methods.

In addition, the subjects were provided with pencil and paper in every session to take notes, if any, pertaining to the problems and to comment on the tutor and its strategies.

Experimental procedure

The experiment was conducted in two phases: training and data collection. In the training phase, three instructional methods were employed to train three groups of subjects to troubleshoot a simulated marine power plant. The effect of training was evaluated in the data collec-

tion phase where all subjects were exposed to identical problems on **Turbinia** without the aid of the tutor.

The subjects were randomly assigned to three groups of ten each. Group I was trained using just **Turbinia**, the simulator. Subjects in Groups II and III were aided by the computer-based tutor **Vyasa**. While **Vyasa** functioned in the passive mode for Group II, it functioned in the active mode for Group III.

Subjects in each of the three groups filled a consent form, completed a survey form and answered questions in Questionnaire 1 prior to the start of the experiment. The subjects were then given written instructional manual appropriate for each group. They were advised to read the instructional manual before starting the first session.

Subjects were distributed into three batches. Each batch was identified by the date on which it started training. Although the assignment of subjects to batches was done randomly, at least three subjects from each group were assigned to every batch. When the subjects from the first batch did not require further assistance in learning the system, a second batch started training on a second computer with fully compatible but slightly slower hardware. After the first batch completed the training and data collection sessions, the second batch was moved to the first computer. Similarly, the third batch started training on the second computer once that machine was available and the experimenter was no longer occupied with the second batch. They moved to the first computer when it became available. Thus, data collection sessions for all batches were done on the same machine. Even though we realized the possible impact of excluding the batching factor in the experimental design, we did not consider it worthwhile to study its effect.

Training phase

The first ten sessions were used for training of subjects in each of the three groups. Each session had a maximum duration of forty-five minutes. The sessions were run on consecutive days with typically one session per day. Occasionally, when a subject missed a day, the lost session was made up by extending the training period by a day. Under no circumstances was a subject permitted multiple sessions in a day.

The first training session for each group introduced the system using a single problem. Audio taped instructions, different for each group, were used during this session. These instructions introduced the subjects to the interface and valid forms of interactions. All interactions between the subject and the interface for this first session were controlled through these instructions. The capabilities of the passive and active tutor were also demonstrated through a predetermined set of actions performed by the subject upon request. The experimenter was present for the entire duration of the first session to answer any questions. Variability of information shared

by the experimenter with the subjects was controlled as far as possible so that consistency could be maintained between subjects and across training groups.

After the first session, subsequent training sessions had three problems each. A subject had a maximum of 13 minutes to solve each problem. If the subject solved the problem early, the next problem was immediately presented. Thus, if the subject solved one or more of the three problems in a session within the allotted time for each problem, the session could potentially be completed in less than 45 minutes.

At the end of each problem the subject was provided the solution. While solutions presented to subjects in the tutor condition were accompanied by an explanation, no such explanation was provided to subjects using the simulator alone.

For the first three training sessions in each group the experimenter was present to answer any questions. For the subsequent sessions the presence of the experimenter in the room was not considered necessary although he was still available to answer questions. It was only on a couple of rare occasions that the subjects sought any clarification from the experimenter past the third session.

At the end of each training session past their third, the subjects were asked to make subjective comments about the instructional system. Although the subjects were free to write anything they were encouraged to identify their "likes" and "dislikes" for the system. At the end of their last training session, the subjects were asked to answer Questionnaire 2.

Data collection phase

The data collection phase consisted of two sessions. These sessions were run on consecutive days immediately following the completion of training. During the data collection sessions, the subjects interacted with the simulator only, unaided by any tutor, irrespective of their training condition.

Each data collection session was approximately 50 minutes long and consisted of 5 problems. If the subject solved the problem within the 10 minute period allocated for each problem, the next problem was presented immediately. However, unlike the training sessions, no solution was provided to the student at the end of each problem. At the end of the data collection sessions, each of the subjects completed an "exit" questionnaire.

Training and test problems

Twenty-four problems were identified for use in the experiment. Since we wished to study the effect of training methods on performance, for familiar as well as novel situations, 5 out of the

24 problems were exclusively reserved for the data collection phase.

For the training phase comprised of 1 single-problem session and 9 three-problem sessions, a total of 28 problems were required. With 5 of the 24 problems reserved for test sessions, there were only 19 available for use in training. Therefore, 9 problems were shown twice to the subjects during training. Selection of these 9 problems was done randomly. However, the same problem was never presented the second time within a span of 3 consecutive sessions.

In the test sessions, in addition to the 5 unseen problems, subjects were given 5 problems from among those seen during training. Three of these problems appeared twice during training and two were seen only once.

The order in which problems were presented during training and test was identical for all groups. In the test sessions, seen and unseen problems were alternated beginning with an unseen problem. The actual order in which the problems appeared was, however, determined randomly.

Performance measures: Dependent variables

Although the ultimate goal in troubleshooting is to successfully identify the failed component, there were several other performance measures considered in the experiment. This section describes the measures used to evaluate the troubleshooter's performance. Subject actions were recorded for computing the performance measures. These measures were obtained directly or derived from the data.

Number of problems solved: Successfully diagnosing a fault is an important measure of troubleshooting ability. Since there is a time limit imposed on solving the problems, a problem is considered solved if a correct diagnosis is made within the time (10 minutes) allocated for each problem in the test sessions.

Solving a problem correctly is not sufficient, however. In real world there are costs associated not only with the troubleshooter's inability to solve problems but also with how the diagnosis was made. Therefore, efficiency of the diagnostic process must also be considered. The remaining performance measures focus on efficiency of troubleshooting.

Troubleshooting time: For those subjects who were successful in solving the problems, the total amount of time taken for solution is a valuable measure of their performance rating. Those who take less time are considered better troubleshooters.

Number of informative actions: A student may take many actions at **Turbinia**'s in-

terface. However, not all actions are informative. Only those actions that are taken to obtain gauge readings are informative since they alone can help the students access the system state information needed to solve the problems. Therefore, the total number of such informative actions provides a measure of the student's overall troubleshooting ability. The smaller the number of informative actions taken to solve a problem, the better is the diagnostic performance.

Percentage of relevant informative actions: Even though every informative action has some information content, only some are directly relevant to the failure. Since the total number of relevant informative actions necessary to solve a problem is dependent on the problem, the percentage of informative actions that are relevant for a failure is a better measure.

Nature of diagnosis: In order to isolate a failed component it is necessary to conduct diagnostic tests that eliminate other probable hypotheses. Due to the limited availability of gauges in the system, the troubleshooter may not be able to isolate the fault completely. However, for each of the failures, there are some gauges that are affected and must be checked to justify pursuing that hypothesis. If a student correctly solves a problem but has not gathered sufficient evidence to do so, the diagnosis is considered premature. On the other hand, if the problem has been seen before then it may not be necessary to gather all evidence before correctly identifying it. However, while attempting to solve a seen problem a student may make several incorrect diagnoses. If these diagnoses suggest hypotheses that are not probable, then the final diagnosis is still considered premature. The rationale for calling such a diagnosis premature is that if a student incorrectly diagnoses a seen problem, then further investigations for that failure should proceed along the lines of an unseen problem.

There may also be times when the student is unable to diagnose the fault even after sufficient evidence implicating the failed component has been gathered. This indicates the student's inability to integrate the diagnostic information and make effective use of it. Such diagnoses are termed as overdue.

Finally, when the student integrates diagnostic information properly, the diagnosis is neither premature or overdue, and hence is considered timely. Categorizing diagnoses as premature, timely or overdue provides a subjective measure of diagnostic performance.

Percentage of guesses: At any time during the troubleshooting process there are likely candidates for the failed component, based on the observed abnormal system states. The likelihood that a component may have failed increases or decreases as more diag-

nostic tests are conducted. While quick diagnosis of a problem saves time and money, incorrect diagnosis costs additional time and money. Even so, selecting a likely component as the cause of abnormal system behavior is not as bad as picking a component that cannot have failed. Thus, an incorrect diagnosis that implicates a component that could not have failed, based on observed symptoms, is a consequence of pure guesswork or inaccurate troubleshooting knowledge. Such an incorrect diagnosis is considered a guess and fewer guesses indicate better troubleshooting performance. Since the number of incorrect diagnoses, in terms of guesses and probable hypotheses, may depend on the problem, percentage of diagnosis for each problem that were guesses is a reasonable measure of troubleshooting performance.

Number of unaffected schematics/subsystems/fluid-paths investigated: For each failure, only a few schematics, subsystems, and fluid-paths are affected. Affected schematics are those that have gauges with abnormal readings. Investigating components in schematics that are unaffected by the failure reflects the student's inability to relate symptoms to the structural location of the power plant. Thus, investigating components in unaffected schematics is undesirable and the number of such schematics wrongly investigated is a measure of performance.

Likewise, investigating unaffected subsystems and fluid-paths reflects the student's inability to relate symptoms to the functional location of the power plant. The number of subsystems and fluid-paths wrongly investigated in this manner are also measures of performance. The fewer the number of unaffected subsystems or fluid-paths investigated by the student in solving a problem, the better is the performance.

The dependent variables listed above are closely related to the troubleshooting strategy that the student is likely to follow. We expected that those trained on the simulator alone would perhaps develop an unguided search strategy to locate the fault. Therefore, they are likely to make more guesses, take a lower percentage informative actions that are relevant, make more premature diagnoses and investigate more unaffected schematics, subsystems and fluid-paths. However, since they would rarely be using concrete reasoning, they would perhaps solve the problem in less time but with many attempts of incorrect diagnosis. Furthermore, it was anticipated that comparing the performance of those trained with the passive and active tutors might reveal individual differences in performance.

In addition to the performance measures described above, a subjective evaluation of the instructional system was also performed. The exit questionnaire (#3) was designed for this purpose. Results of analyzing this questionnaire and an analysis of the data collected from the experiment are discussed in the next section.

EXPERIMENTAL RESULTS

Analysis of the experimental data revealed that subjects in the unaided group developed a troubleshooting strategy distinctly different from those of subjects that were aided by **Vyasa** during training. The unaided group did not devise any good and consistent troubleshooting strategy and often relied on unguided search for the failure. As such, they conducted a large number of diagnostic tests and very few of these tests were relevant to the failure. In the absence of a guided strategy and in their pursuit for a quick diagnosis, the unaided group provided a large number of incorrect diagnoses and investigated a large number of unaffected schematics, subsystems and fluid paths.

On the other hand, the two groups aided by the tutor hypothesized probable failures and conducted diagnostic tests to either strengthen or weaken their hypotheses. Most of the informative actions they took were relevant to the failure. In spite of an incentive to solve problems quickly, the aided groups did not indulge in much guesswork. Most of their guesses appeared to result from panic that set in when they were running out of time. Also, since their investigations were more focussed on hypotheses that explained observed abnormal behavior, the aided groups investigated fewer unaffected schematics, subsystems and fluid paths.

While subjects in each group solved approximately the same number of test problems, on an average the unaided group solved problems in shorter time. This was not at all surprising since this group relied heavily on guessing. Thus, their good performance based on quick diagnosis of failures was offset by the numerous incorrect diagnoses for each problem.

For both the aided and unaided groups, performance on most measures was better with seen than unseen problems. Among seen problems, those seen twice were often better recalled by all groups, indicating that practice helped subjects develop symptom-cause associations. The two aided groups performed better than the unaided group with unseen problems indicating that the training they received was successfully transferred to unfamiliar situations as well.

There were slight differences in the performance between the two aided groups. Those trained with the active tutor made a smaller number of guesses and hence fewer premature diagnoses, but they also solved a smaller number of problems. Most of the variation in the performance between the two aided groups seems to have been caused by individual differences. In the active tutor group, there were a few subjects who became somewhat dependent on the tutor; their performance suffered once the aid was withheld in the test sessions. There were others in both aided groups that became overly conservative and often investigated and eliminated less likely alternatives as well. This may not necessarily be a bad troubleshooting strategy, since incorrect

diagnosis are costly. However, any delay in diagnosing the fault could also be costly. Such accuracy-time trade-offs stem more due to individual preferences rather than the instructional strategy used for training.

Discussion of results

From an analysis of the results, it is clear that the tutor in both the passive and the active modes helped the students to develop useful troubleshooting strategies. The tutor was also helpful for forming plausible failure hypotheses based on observed symptoms and for systematically eliminating them by conducting appropriate diagnostic tests. In comparison, those trained without the tutor did not develop good troubleshooting strategies. They relied rather heavily on guessing the solution. Furthermore, the tutor helped the students to recognize and integrate crucial diagnostic information in a timely manner that the students without the tutor were unable to do. Students trained by the tutor were better prepared for unfamiliar situations than those trained on the simulator.

The data also indicated that the effectiveness of a tutoring strategy depended upon the individual student. For example, the strategy of providing explanations for all observed symptoms for each problem was intended to help the students develop a proper causal model of fault propagation. Some students who learned to map salient symptoms to causes from these explanations became overly conservative. During troubleshooting they spent a lot of time eliminating all probable hypotheses linked to an observed symptom even when sufficient evidence in support of a highly probable hypothesis had been collected. Another tutoring strategy adopted by the active tutor was to provide help in building, refining, and eliminating failure hypotheses. In this capacity the active tutor came to be perceived by some students as an on-line associate. These students often took the help of the active tutor to refine their failure hypotheses and thus became dependent on the tutor to solve problems. Performance of these students deteriorated when the active tutor was withdrawn.

In summary, results of the experiment show that a simulator alone is inadequate. A simulator in conjunction with a computer-based tutor can help develop efficient troubleshooting skills. However, not all students are equally receptive to every tutoring strategy and provisions must be made in training programs for individual preferences and differences in abilities and styles. A more complete and thorough analysis of the experimental data and results will appear in Vasandani (1991).

CONCLUSIONS

The research program described in this report has contributed to the field of training for diagnostic problem solving in realistic, complex dynamic system domains. We have shown the viability of designing and implementing an effective intelligent tutoring system for supervisory control operation. Instructional systems that integrate an ITS with a simulator and provide access via direct manipulation graphical interfaces can contribute greatly to an effective training program. We hope that our research results will stimulate further research and development of tutoring and training systems and implementation within the navy and elsewhere. As a result of our research, we have gained a better understanding of diagnostic problem solving in complex engineering domains. In addition, we have developed good insights and expertise in modeling large dynamic systems, knowledge representation for tutoring, and interactive interfaces. To benefit from our experience and expertise, we are planning a program of research to develop models of expertise that will further enhance tutoring and training systems and intelligent operator associates.

ACKNOWLEDGMENTS

We wish to thank the commanders and the officers of the Georgia Tech Naval ROTC unit for their cooperation in providing access to their cadets and for technical help. Lt. William A. Marriot was generous with his time and helped us with many technical matters. We thank the thirty volunteer cadets for participating as subjects in our experiment. We are grateful to Dr. Susan Chipman, Contract Monitor, for her critical comments and suggestions on methodological issues relevant to cognitive science and training throughout the duration of the project and for her efforts to make our research known to various naval units. The comments and suggestions of Drs. S. Collyer, H. W. Sinaiko, and W. Vaughn at briefings during various stages of the project were helpful. Dr. Christine Mitchell at Georgia Tech deserves our thanks for her help with many design and implementation issues and with statistics.

REFERENCES

Burstall, A. F. (1965). *A history of mechanical engineering*. Cambridge, MA: MIT Press.

Clancey, W. J. (1987). *Knowledge-based tutoring: the GUIDON program*, Cambridge, MA: MIT Press.

Fath, J. L., Mitchell, C.M., & Govindaraj, T. (1990). An ICAI architecture for troubleshooting in complex, dynamic systems. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-20(3)*, 537-558.

Govindaraj, T. (1987). Qualitative approximation methodology for modeling and simulation of large dynamic systems: applications to a marine steam power plant. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-17*(6), 937-955.

Govindaraj, T., & Su, Y.-L. (1988). A model of fault diagnosis performance of expert marine engineers. *International Journal of Man-Machine Studies, 29*, 1-20. (Also in In J. Boose, & B. Gaines (Ed.) (1989), *The foundations of knowledge acquisition*. London: Academic Press.)

Hollan, J. D., Hutchins, E. L., & Weitzman, L. (1984). STEAMER: An interactive inspectable simulation-based training system. *AI Magazine*, (Summer 1984), 15-27.

Lesgold, A., Lajoie, S., Bunzo, M., & Eggan, G. (1989). SHERLOCK: A coached practice environment for an electronics troubleshooting job. In J. Larkin, R. Chabay, and C. Scheftic (Ed.), *Computer assisted instruction and tutoring systems: Establishing communication and collaboration*, Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Miller, J. R. (1988). Human-Computer interaction and intelligent tutoring systems. In Polson and Richardson, (Ed.), *Foundations of ITSs*, Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Miller, R. A. (1985). A systems approach to modelling discrete control performance. In W. B. Rouse, (Ed.), *Advances in Man-Machine Systems Research*, Volume 2, Greenwich, CT: JAI press.

Moran, T. P. (1983). Getting into a system: external-internal task mapping analysis. Proceedings of the ACM-CHI Conference on Human Factors in Computing Systems. Boston, 45-49.

Psotka, J., Massey, L. D., & Mutter, S. A. (Ed.). *(1988). Intelligent tutoring systems: Lessons learned.* Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Rasmussen, J. *(1986). Information processing and human machine interaction: An approach to cognitive engineering,* New York, NY: North- Holland.

Sleeman, D., & Brown, J. S., (Ed.). (1982). *Intelligent tutoring systems,* Orlando, FL: Academic Press.

Su, Y.-L. (1985). *Modeling fault diagnosis performance on a marine power plant simulator.* Doctoral dissertation, Georgia Institute of Technology

Su, Y.-L., & Govindaraj, T. (1986). Fault Diagnosis in a Large Dynamic System: Experiments on a Training Simulator. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-16*(1), 129-141.

Towne, D. M., & Munro, A. (1988). The intelligent maintenance training system. In J. Psotka, J. L. D. Massey, & S. A. Mutter (Ed.), *Intelligent tutoring systems: Lessons learned* (pp. 479-530). Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Vasandani, V., & Govindaraj, T. (1989). An intelligent tutor for diagnostic problem solving in complex dynamic systems. Boston, Massachusetts: *Proceedings of the 1989 International Conference on Systems, Man, and Cybernetics,* 772-777.

Vasandani, V. (1991). *Intelligent tutoring for diagnostic problem solving in complex dynamic systems.* Doctoral dissertation in progress, Georgia Institute of Technology.

Vasandani, V., & Govindaraj, T. (1990). Knowledge representation and human-computer interaction in an intelligent tutor for diagnostic problem solving. *Proceedings of the 1990 International Conference on Systems, Man, and Cybernetics,* Los Angeles, California, 665-667.

Wenger, E. (1987). *Artificial intelligence and tutoring systems: computational and cognitive approaches to the communication of knowledge,* Los Altos, California: Morgan Kaufmann.

Woods, D. D. (1984). Visual momentum: a concept to improve the cognitive coupling of person and computer. *International Journal of Man-Machine Studies, 21,* 229-244

Technical Report Distribution List - Manpower, Personnel, and Training R&D Program

As of 11/1/90
One copy to each addressee except as noted:

Director, Contract Research Department
Office of Naval Research (Code 11)
Arlington, VA 22217-5000

Chairman, MPT R&D Committee
Office of the Chief of Naval Research
Code 222
Arlington, VA 22217-5000

Program Manager, Operations
Research (Code 1111MA)
Office of Naval Research
Arlington, VA 22217-5000

Director, Life Sciences (Code 114)
Office of Naval Research
Arlington, VA 22217-5000

Director, Cognitive & Neural Sciences
(Code 1142)
Office of Naval Research
Arlington, VA 22217-5000

Cognitive Science (Code 1142CS)
Office of Naval Research
Arlington, VA 22217-5000

Perceptual Science (Code 1142PS)
Office of Naval Research
Arlington, VA 22217-5000

Defense Technical Information Center*
DTIC/DDA-2
Cameron Station, Building 5
Alexandria, VA 22314

CDR J. S. Hanna, Office of the Deputy
Asst. Secretary of the Navy (Manpower)
5D800, The Pentagon
Washington, DC 20350-1000

Head, Manpower, Personnel, and
Training Branch
Office of the CNO (Op-813)
4A478, The Pentagon
Washington, DC 20350-1000

Assistant for Manpower and Training
Office of the CNO (Op-911H)
5D772, The Pentagon
Washington, DC 20350-2000

Assistant for Planning and Technology
Development
Office of the DCNO(MPT) (Op-01B2)
Department of the Navy, AA-1822
Washington, DC 20350-2000

Deputy Director Total Force Training
and Education Division
Office of the DCNO(MPT) (Op-11B)
Department of the Navy
Washington, DC 20370-2000

R&D Coordinator, Attn: Jan Hart
Office of the DCNO(MPT) (Op-11K1)
Department of the Navy, AA-G817
Washington, DC 20370-2000

Deputy Director Military Personnel
Policy Division
Office of the DCNO(MPT) (Op-13B)
Department of the Navy, AA-1825
Washington, DC 20370-2000

Head, Military Compensation Policy
Branch
Office of the DCNO(MPT) (Op-134)
Department of the Navy, AA-2837
Washington, DC 20370-2000

*Note: 12 copies go to DTIC

Headquarters U.S. Marine Corps
Code MA
Washington, DC 20380-0001

Head, Leadership Branch
Naval Military Personnel Command
Attn: LT Gary Kent, NMPC-621
Department of the Navy, AA-1603
Washington, DC 20370-5620

Director, Research & Analysis Division
Navy Recruiting Command (Code 223)
4015 Wilson Boulevard, Room 215
Arlington, VA 22203-1991

Technical Director Silva
Attn: Dr. Kobus
Naval Health Research Center
P.O. Box 85122
San Diego, CA 92138-9174

Head, Human Factors Division
Naval Training Systems Ctr. (Code 26)
12350 Research Parkway
Orlando, FL 32826-3224

Naval Training Systems Center
ATTN: Dr. Eduardo Salas, (Code 262)
12350 Research Parkway
Orlando, FL 32826-3224

Naval Training Systems Center
ATTN: Dr. Robert Hays, (Code 262)
12350 Research Parkway
Orlando, FL 32826-3224

Technical Director
Navy Personnel R&D Center
Code 01
San Diego, CA 92152-6800

Director, Manpower Systems Dept.
Code 11
Navy Personnel R&D Center
San Diego, CA 92152-6800

Director, Personnel Systems Dept.
Code 12
Navy Personnel R&D Center
San Diego, CA 92152-6800

Director, Testing Systems Department
Navy Personnel R&D Center
Code 13
San Diego, CA 92152-6800

Director, Training Systems Dept.
Code 14
Navy Personnel R&D Center
San Diego, CA 92152-6800

Director, Training Technology Dept.
Code 15
Navy Personnel R&D Center
San Diego, CA 92152-6800

Director, Organizational Systems Dept.
Code 16
Navy Personnel R&D Center
San Diego, CA 92152-6800

Naval Ocean Systems Center
Command Support Technology Division
Attn: Mr. Jeffrey Grossman, Code 4402
San Diego, CA 92152-5000

Chairman, Dept. of Admin. Sciences
(Code AS)
Naval Postgraduate School
Monterey, CA 93943-5100

Chairman, Department of Operations
Research (Code OR)
Naval Postgraduate School
Monterey, CA 93943-5100

Director, Instructional Development and
Educational Program Support Dept.
Naval Education and Training Program
Management Support Activity
(NETPMSA)
Pensacola, FL 32509-5100

Academic Programs and Research
Branch
Naval Technical Training Command
Code N6, NAS Memphis, Bldg. C-1
Millington, TN 38054-5056

Director, Defense Personnel Security
Research and Education Center
Suite E, Building 455
99 Pacific Street
Monterey, CA 93940-2481

Technical Director
U.S. Army Research Institute for the
Behavioral and Social Sciences
5001 Eisenhower Avenue
Alexandria, VA 22333-5600

Chief Scientist
Air Force (AFHRL)
Brooks AFB, TX 78235

Director, Manpower & Training Program
Center for Naval Analyses
4401 Ford Avenue
P.O. Box 16268
Alexandria, VA 22302-0268

Library- Code 231
Navy Personnel R&D Center
San Diego, CA 92152-6800

Library
Naval War College
Newport, RI 02940

Chief, Survey and Market
Analysis Division
Defense Manpower Data Center
1600 Wilson Boulevard, #400
Arlington, VA 22209-2593

Program Director
Manpower Research & Advisory Services
Smithsonian Institution
801 North Pitt Street, Suite 120
Alexandria, VA 22314-1713